Conf. univ. dr. Gabriela MEŞNIȚĂ

ANALIZA ŞI PROIECTAREA SISTEMELOR INFORMAȚIONALE I

Cap. 6 Modelarea conceptuală a datelor: diagrama entitate-relație[1]

6.1 Rolul modelării datelor 2
6.2 Definirea conceptelor utilizate în construirea DER 3
6.2.1 Identificarea şi descrierea entităților de date 3
6.2.2 Definirea relațiilor dintre entitățile de date 4
6.2.3 Cardinalitatea relațiilor 6
6.2.4 Construirea diagramelor entitate-relație 9
6.2.5 Notații folosite pentru construirea DER 11
Notația Chen 11
Notația Ross 11
Notația LBMS 12

Iaşi, 2004

[1] Capitol realizat pe baza lucrărilor:
Dumitriu, F. – *Proiectarea sistemelor informatice*, Suport curs, Univ. "Al. I. Cuza" Iași, an univ. 2002/2003
Oprea, D. – *Analiza şi proiectarea sistemelor informaționale economice*, Ed. Polirom, Iași, 1999, pp. 195-224
Modell, M. – A Professional's Guide to Systems Analysis, Second Edition, McGraw-Hill Book Company, New York, 1996, pp. 185-206
Teorey, T.J., *Database Modeling & Design*, Morgan Kaufmann Publishers, Inc, 1999, pp. 44-47
Langer, A.M. – *Analysis and Design of Information Systems*, Second Edition, Springer, New York, 2001, pp. 44-64
Marakas, G.M. – *Systems Analysis and Design: An Active Approach*, Prentice Hall, Upper Saddle River, New Jersey, 2001, pp. 150-172
Oprea, D., Dumitriu, F., Meşniță, G. – *Proiectarea asistată de calculator a sistemelor informatice*, Ed. Universității „Al. I. Cuza" Iași, 1998, pp. 37-58
Pressman, R.S. – *Software Engineering. A Practitioner's Approach*, European adaptation, Fifth Edition, McGraw-Hill, London, 2000, pp. 293-328

Prin *modelarea conceptuală a datelor* se urmăreşte construirea unui model al datelor care să asigure transpunerea exactă a realităţii din domeniul analizat, fără a lua în considerare cerinţele specifice unui model de organizare a datelor (cum este modelul relaţional), criteriile de calitate privind organizarea datelor, cerinţele nefuncţionale ale sistemului şi criteriile de performanţă privind stocarea şi accesarea datelor. În acest sens, se construieşte diagrama entitate-relaţie, care evidenţiază entităţile de date din sistem, atributele acestora, precum şi legăturile dintre entităţi. Modul în care vor fi implementate legăturile dintre entităţi, de exemplu, nu interesează în acest moment, atenţia fiind îndreptată doar spre identificarea şi descrierea lor.

6.1 Rolul modelării datelor

Introducerea modelului entitate-relaţie (ER) a apărut ca răspuns la complexitatea proiectării logice a bazelor de date de dimensiuni mari. Argumentarea acestei modalităţi de abordare a datelor constă în faptul că în proiectarea tradiţională a bazelor de date, în special a celor relaţionale, presupune constituirea relaţiei universale prin reunirea tuturor datelor elementare (atribute) identificate în faza de analiză şi repartizarea lor în tabele pe baza analizei dependenţelor dintre atribute (dependenţe funcţionale, dependenţe multivaloare şi de joncţiune) şi aplicarea procesului de normalizare. Abordarea se pretează foarte bine în cazul bazelor de date de dimensiuni mici şi medii, dar devine destul de greoaie în cazul bazelor de date de dimensiuni mari şi foarte mari.

Modelarea conceptuală a datelor cu ajutorul diagramelor entitate-relaţie (DER) a fost descrisă prima dată în lucrările lui P.P. Chen, publicate în 1976, deşi primele încercări de formalizare sunt înregistrate în anii '60 şi aparţin lui Charles Bachman. Ulterior, modelul lui Chen a înregistrat numeroase modificări şi extensii. Simplitatea, uşurinţa învăţării şi posibilitatea formalizării cerinţelor sistemului aşa cum sunt ele în realitate, independent de opţiunile de organizare şi tehnologice au sporit vertiginos popularitatea modelului ER încă din anii '80.

Noua abordare presupune, mai întâi, modelarea conceptuală a datelor prin construirea diagramei entitate-relaţie (DER), care evidenţiază entităţile de date ale sistemului, proprietăţile acestora şi legăturile dintre entităţi. Ulterior, prin aplicarea unor reguli simple, are loc transformarea modelului entitate-relaţie în schema logică a bazei de date. Tabelele astfel obţinute sunt, în final, analizate din perspectiva normalizării putând rezulta noi tabele.

Utilizarea modelului ER oferă o serie de avantaje faţa de abordarea tradiţională[2]:

- *reprezintă un util instrument de comunicare* între echipa de dezvoltare a sistemului (analişti şi proiectanţi) şi utilizatorii finali pe parcursul fazelor de analiză şi proiectare conceptuală şi logică;
- *este uşor de înţeles şi conceput*. Ca şi în cazul modelării proceselor, prezentarea grafică permite exprimarea unui volum mare de informaţii sub o formă compactă, uşor de urmărit şi înţeles;
- *apelează la abstractizare*, ceea ce reduce considerabil numărul obiectelor luate în considerare la analiza şi proiectarea bazei de date. Prin utilizarea noţiunii de entitate ca abstractizare pentru datele elementare (atribute) se vor analiza mai puţine obiecte (numărul entitaţilor de date este mult mai mic decât numărul datelor elementare din sistem) şi mai puţine relaţii între obiecte (numărul relaţiilor dintre entităţi este mult mai mic decât numărul relaţiilor de dependenţă existente între atribute). Deşi datele elementare sunt reprezentate şi în această abordare, ca proprietăţi ale entităţilor, numărul dependenţelor ce trebuie analizate este mult redus, fiind luate în considerare doar cele stabilte la nivelul entităţilor şi nu la nivelul tuturor atributelor din relaţia universală pe baza căreia se construieşte baza de date;
- *existenţa unui set complet de reguli de transformare a DER în tabele ale bazei de date*. Aceste reguli permit transformarea simplă şi rapidă a cerinţelor informaţionale ale

[2] adaptare după Teorey, T.J., *Database Modeling & Design*, Morgan Kaufmann Publishers, Inc, 1999, p.46

sistemului, structurate în DER, în baza de date. Majoritatea instrumentelor CASE oferă suport pentru generarea automată a bazei de date în funcție de SGBD-ul dorit.

6.2 Definirea conceptelor utilizate în construirea DER

Analiza cerințelor informaționale ale noului sistem reprezintă etapa cea mai importantă care stă la baza proiectării bazei de date. Înglobarea acestor cerințe în schema finală a bazei de date reprezintă o provocare pentru proiectanți. Obiectivele majore ale analizei cerințelor din perspectiva modelării datelor vizează[3]:

- descrierea cerințelor de date ale sistemului în termenii entităților de date;
- colectarea informațiilor care descriu entitățile de date şi relațiile dintre acestea care să permită construirea DER;
- determinarea tranzacțiilor ce vor fi efectuate asupra bazei de date şi descrierea interacțiunii dintre tranzacții şi entitățile de date;
- definirea cerințelor nefuncționale ale bazei de date, cum ar fi cele legate de performanțe, integritate, securitate, administrarea datelor;
- specificarea platformei hardware şi software pe care va fi implementată baza de date.

După formularea clară a cerințelor se trece la formalizarea acestora, respectiv construirea DER, care are la bază trei clase de obiecte: entități, atribute şi relații, şi presupune parcurgerea a doi paşi:

- identificarea şi descrierea entităților şi
- definirea relațiilor dintre entități.

6.2.1 Identificarea şi descrierea entităților de date

Prin entități de date se face referire la obiectele despre care este nevoie să se memoreze informații. De regulă, ele se referă la locuri, persoane, lucruri, evenimente în legătură cu care există interes din perspectiva stocării datelor. Un caz particular al unei entități este numit *instanță* sau *ocurență*. Astfel, o entitate va conține mai multe instanțe de acelaşi tip, adică instanțe cu proprietăți comune. Unii autori utilizează noțiunile de clasă de entități şi entitate pentru entitate şi, respectiv, instanță.

Exemple de entități sunt: *angajat*, care va avea drept instanțe toți angajații dintr-o firmă; *furnizor*, care va conține toți partenerii de afaceri cu care firma are relații de aprovizionare; *consum*, care va avea drept instanțe toate documentele de consum înregistrate într-o perioadă de timp în firmă. Alte exemple pot fi: departament, produs, student, disciplina, profesor, localitate etc.

O entitate este descrisă prin intermediul *atributelor*. Atributele reprezintă proprietăți ale entității pe care o descriu, toate instanțele unei entități fiind descrise de aceleaşi atribute. Un atribut al unei instanțe dintr-o entitate se numeşte *valoarea atributului*. De exemplu, atributele care caracterizează entitatea angajat pot fi: marca, numele, adresa, numărul de telefon, CNP, funcția, salariul etc.

Există două tipuri de atribute: *identificatori* şi *descriptori*. Un identificator (numit şi cheie) are rolul de a identifica în mod unic fiecare instanță care poate popula o entitate de date. În schimb, descriptorii specifică caracteristici ne-unice ale instanțelor unei entități.

Depistarea entităților unui sistem nu reprezintă o activitate tocmai uşoară. În multe situații analistul trebuie să decidă dacă un obiect reprezintă o entitate, un atribut al unei entități sau o relație între entități. Să luăm drept exemplu obiectul *localitate*; el reprezintă o entitate sau un atribut? În sistemul de aprovizionare, el ar putea reprezenta o entitate sau doar atributul entității furnizor. Un alt exemplu este *comanda*; reprezintă comanda o entitate sau o relație între entitățile *Material* şi *Furnizor*?

La clasificarea entităților şi atributelor trebuie respectate următoarele două reguli:

[3] Teorey, T.J., *Database Modeling & Design*, Morgan Kaufmann Publishers, Inc, 1999, p.46-47

- *entitățile trebuie să conțină informații descriptive*. Conform acestei reguli, dacă pentru un obiect există informații descriptive (adică mai multe atribute care să o caracterizeze) atunci acel obiect constituie o entitate. Altfel, dacă pentru obiectul respectiv este necesar doar un identificator, nefiind solicitate atribute de tip descriptori, atunci acel obiect va reprezenta un atribut. Dacă pentru obiectul *Localitate* sunt necesare mai multe informații descriptive (cum ar fi județul, regiunea, numărul populației etc.) atunci el trebuie clasificat ca entitate. Dacă se solicită doar numele localității, atunci el va fi doar un atribut al unei entități (cum ar fi furnizor, angajat, departament etc.).
- *un atribut trebuie ataşat acelei entități pe care o descrie în modul cel mai direct*. De regulă, o proprietate se referă la o singură entitate, deci un atribut va fi regăsit doar la o singură entitate. În sistemul de aprovizionare, de exemplu, entitatea *Comandă* nu va conține nici numele furnizorului (sau codul acestuia), nici denumirea materialului (sau codul materialului); aceste atribute caracterizează mai direct entitățile *Furnizor* şi, respectiv, *Material*. Nu intră sub incidența acestei reguli sinonimele. Un exemplu în acest sens poate fi atributul *adresa*: el poate caracteriza atât entitatea *angajat*, cât şi entitatea *furnizor*.

Aşadar, clasificarea unui obiect ca entitate sau atribut presupune analiza cu multă atenție a documentației sistemului în care sunt formulate şi detaliate cerințele informaționale.

6.2.2 Definirea relațiilor dintre entitățile de date

O relație reprezintă legătura care există în lumea reală între una, două sau mai multe entități. Relațiile nu au o existență fizică sau conceptuală, ci depind de entitățile asociate. Altfel spus, existența relațiilor depinde de existența entităților asociate (în schimb existența unei entități nu depinde de existența unei relații cu o altă entitate). Un caz particular al unei relații se mai numeşte *instanța relației* sau *ocurența relației*. Instanța relației se referă la legătura dintre instanțele entităților asociate. Exemple de relații sunt: *emite*, între entitățile *Factura* şi *Furnizor*; *conține*, între entitățile *Factura* şi *Produs*; *este condus*, între entitățile *Departament* şi *Angajat* sau orice verb care care descrie conexiunea dintre entități.

Descrierea unei relații presupune specificarea următoarelor elemente: ordinul (gradul), cardinalitatea, rolul şi eventualele atribute care o caracterizează. În continuare ne vom opri pe rând asupra fiecărui element descriptiv.

Ordinul relației este dat de numărul entităților asociate printr-o relație. Conform acestui criteriu, relațiile pot fi clasificate în:

- *relații recursive*, numite şi relații ale entității cu ea însăşi. Acest tip de relație leagă două instanțe ale aceleiaşi entități. Un exemplu de relație recursivă este redat în figura 6.1.

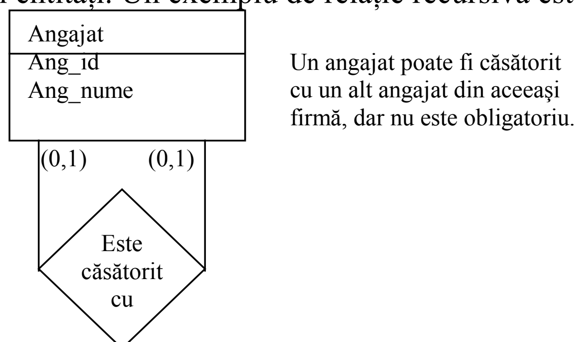
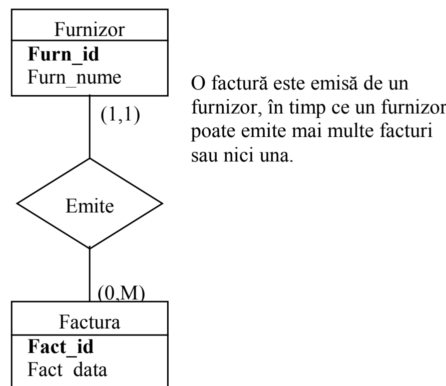
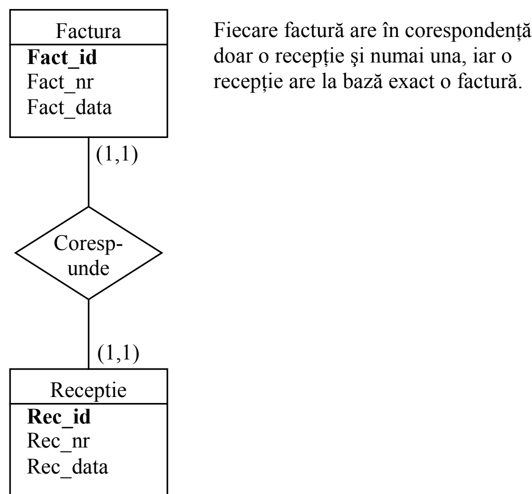


Fig. 6.1 Relație recursivă

- *relații binare*, în care sunt implicate două entități. Acest tip de relații sunt de departe cele mai întâlnite în lumea reală. Mai mult, unele metode utilizează doar acest tip de relații la construirea modelului ER. Exemple de relații binare se regăsesc în următoarea figură:



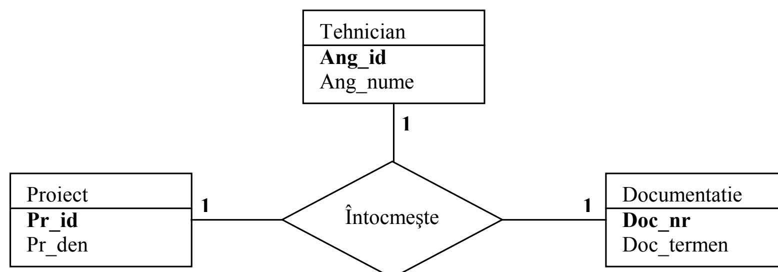
a) Exemplu 1 de relație binară



b) Exemplu 2 de relație binară

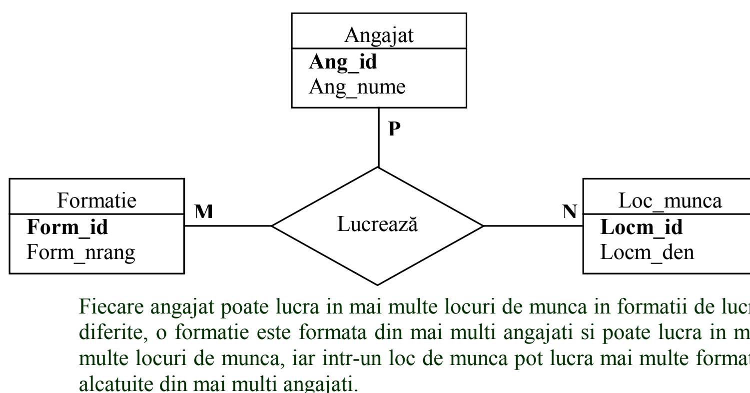
Fig. 6.2 Relații binare

- *relații de ordinul n*, numite şi relații **n**-are. Acest tip de relații se stabileşte între trei sau mai multe entități. Dacă numărul entităților implicate într-o relație este 3, atunci vom avea o relație ternară. Astfel de exemple sunt prezentate în figura 6.3. Relațiile ternare sunt mai rar întâlnite în lumea reală, ele fiind utilizate atunci când nu este suficientă utilizarea relațiilor binare pentru descrierea cu acuratețe a semanticii relației respective.



Un tehnician poate lucra la mai multe proiecte, însă el intocmeşte o singură documentație pentru fiecare proiect în parte; fiecare documentație este întocmită de un singur tehnician pentru un singur proiect; un proiect poate avea mai multe documentații, dar fiecare documentație va fi intocmită de un singur tehnician.

a) Exemplu 1 de relație ternară



b) Exemplu 2 de relaţie ternară

Fig. 6.3 Relaţii ternare

6.2.3 Cardinalitatea relaţiilor

Următorul element utilizat pentru descrierea relaţiilor îl reprezintă *cardinalitatea*. Cardinalitatea unei relaţii defineşte numărul instanţelor unei entităţi care pot fi asociate unei instanţe a celeilalte entităţi prin intermediul relaţiei considerate. Pentru o relaţie dată trebuie specificată cardinalitatea pentru fiecare entitate asociată. Cardinalitatea relaţiei pentru o entitate este sugerată printr-o pereche de valori, în cazul relaţiilor binare trebuind specificate două astfel de perechi de valori. Valorile care compun o astfel de pereche semnifică:

- *cardinalitatea minimă*, pentru care sunt posibile valorile „0” sau „1”, şi
- *cardinalitatea maximă*, pentru care sunt posibile valorile „1” sau „M” (adică „multe”).

Să luăm drept exemplu relaţia *emite/este emis* dintre entităţile *Furnizor* şi *Factura*, prezentată în figura 6.4. Cardinalitatea minimă a relaţiei pentru entitatea *Factura* este „1”, iar cea maximă este tot „1”; cardinalitatea relaţiei pentru entitatea *Furnizor* este „0”, respectiv „M”. Prin urmare, în relaţia *emite/este emis*, unui furnizor (care este o instanţă a entităţii *Furnizor*) îi poate corespunde minim 0 şi maxim „multe” facturi (adică instanţe ale entităţii *Factura*), în timp ce unei facturi îi corespunde minim un furnizor şi maxim tot un furnizor. Dacă luăm în considerare şi numele relaţiei (sau rolurile entităţilor în cadrul relaţiei), atunci o relaţie poate fi citită în ambele sensuri astfel:

„un furnizor emite 0 sau mai multe facturi”, respectiv
„o factură este emisă de un furnizor şi numai unul”.

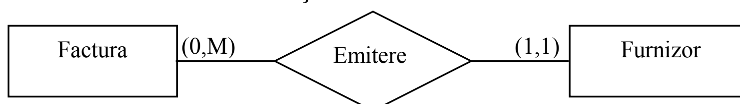


Fig. 6.4 Cardinalitatea relaţiilor

Anterior am prezentat clasificarea relaţiilor în funcţie de gradul lor. Un alt criteriu de clasificare a relaţiilor este legat de cardinalitatea maximă. Astfel, în cazul relaţiilor recursive şi al celor binare vom avea trei tipuri de relaţii:

- *relaţii de tipul 1:1 (unu-la-unu)*, adică o instanţă a entităţii X este asociată unei singure instanţe a entităţii Y şi reciproc. Exemple se regăsesc în figurile 6.2 b), dar şi în figura 6.5.
- *relaţii de tipul 1:N (unu-la-multe),* însemnând că o instanţă a entităţii X poate fi asociată la *n* instanţe ale entităţii Y, în timp ce o instanţă a entităţii Y va fi asociată doar unei singure instanţe a entităţii X. Acest tip de relaţii este cel mai des întâlnit în lumea reală, exemple fiind prezentate în figura 6.2 a), dar şi în figura 6.6.
- *relaţii de tipul M:N (multe-la-multe)*, care presupun că orice instanţă a entităţii X poate fi asociată mai multor instanţe ale entităţii Y, iar o instanţă a entităţii Y poate de asemenea să aibă asociate mai multe instanţe ale entităţii X (figura 6.7).

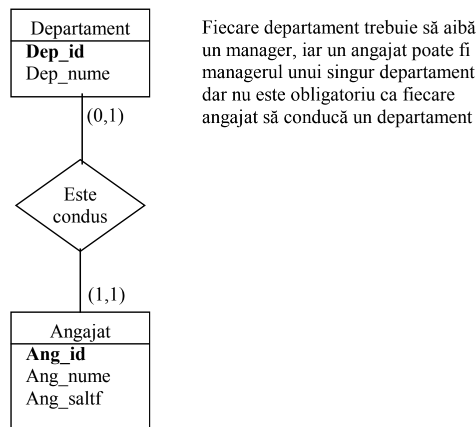


Fig. 6.5 Exemplu de relație 1:1

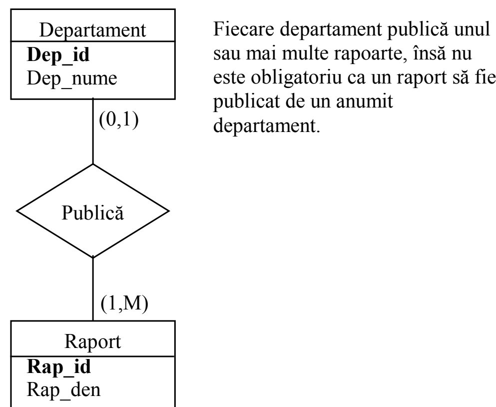


Fig. 6.6 Exemplu de relație 1:N

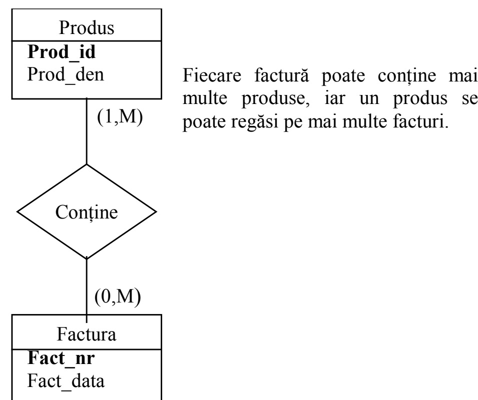


Fig. 6.7 Exemplu de relație M:N

Generalizând, se poate afirma că o relație de ordinul n va înregistra n+1 tipuri de relații din punctul de vedere al cardinalității maxime. Dacă, după cum deja am văzut, în cazul unei relații de ordinul 2 avem 3 tipuri de relații, pentru o relație de ordinul 3 (relație ternară) vom avea 3+1=4 tipuri de relații: **1:1:1** (fig. 6.3 a)), **1:1:P**, **1:N:P** (fig. 6.8) și **M:N:P** (fig. 6.3. b)).

S-a discutat până acum importanța cardinalității maxime, însă și *cardinalitatea minimă* are o semnificație importantă în proiectarea bazelor de date, ea fiind legată de conceptul de *restricție de dependență*[4]. Dacă existența instanței x din entitatea X depinde de existența instanței y din entitatea Y, se spune că x este dependentă de y; de aici rezultă că ștergerea entității y atrage automat ștergerea entității x din baza de date. Se spune că existența instanței x depinde de existența instanței y. Entitatea Y este denumită entitate dominantă, iar entitatea X este denumită entitate dependentă.

[4] Korth, H.F., Silberschatz, A., *Sistemes de gestion de bases de donees*, McGraw-Hill, Paris, 1988, citat în Fotache, M., *Baze de date relaționale. Organizare și interogare*, Ed. Junimea, Iași, 1996, p. 69

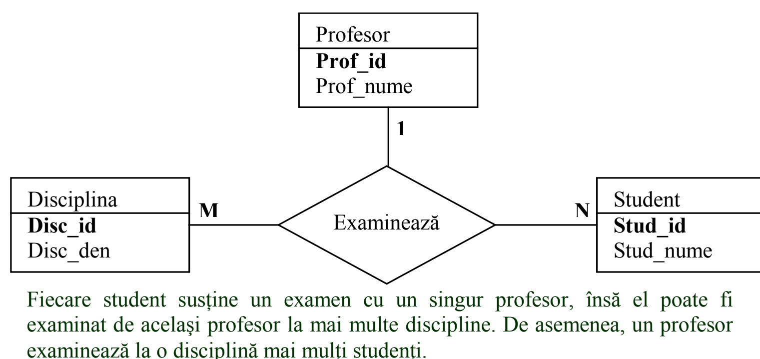


Fig. 6.8 Exemplu de relaţie 1:N:P

Revenind la exemplul anterior cu relaţia *emite/este emisa*, entitatea *Factura* este dependentă de entitatea *Furnizor*, deci *Furnizor* este o entitate dominantă. Ştergerea unei facturi nu atrage automat şi ştergerea furnizorului care a emis-o, deoarece este posibil ca în baza de date să mai rămână facturi primite de la furnizorul respectiv. În schimb, ştergerea unui furnizor atrage după sine ştergerea tuturor facturilor aferente acestuia.

Relaţiile dintre entităţi pot fi clasificate după cardinalitatea minimă în două tipuri: *relaţii facultative* şi *relaţii obligatorii*. Altfel spus, participarea unei entităţi într-o relaţie poate fi facultativă sau obligatorie; participarea opţională este pusă în evidenţă prin cardinalitatea minimă „0" pentru o entitate, iar cardinalitatea minimă „1" semnifică faptul că acea entitate este obligatoriu să participe într-o relaţie. Se observă că în relaţia *emite/este emisă* entitatea *Factura* are cardinalitatea minimă „1", deci este obligatorie participarea oricărei instanţe (facturi) într-o relaţie, în timp ce cardinalitatea minimă pentru entitatea *Furnizor* este „0", ceea ce înseamnă că nu este obligatorie participarea fiecărui furnizor în relaţia *emite/este emisă* (va putea exista un furnizor care să nu participe în nici o relaţie).

Rolul defineşte funcţia care atrage entitatea într-o relaţie. Pentru o relaţie trebuie specificat rolul fiecărei entităţi asociate. De exemplu, în relaţia dintre *Factura* şi *Furnizor* exemplificată în figura 6., rolul entităţii Furnizor este *Emite*, iar cel al entităţii Factura este *este emisă*. Specificarea rolurilor jucate de entităţi în DER ajută la clarificarea aspectelor semantice ale modelării datelor, precum şi la citirea relaţiilor, după cum s-a văzut anterior. Prin combinarea rolurilor jucate de entităţile asociate se obţine numele relaţiei (emite/este emisă). Totuşi, unele metode de recomandă utilizarea unui singur nume pentru o relaţie, stabilit prin alegerea unui substantiv care să reflecte logica legăturii dintre entităţile respective. De exemplu, pentru relaţia din figura 6.4 se poate utiliza numele *Cumpărare* sau *Emitere*. În această variantă, rolul fiecărei entităţi nu mai este specificat în mod explicit, însă el rezultă implicit din numele atribuit relaţiei. În lipsa unui substantiv semnificativ pentru logica relaţiei se poate alege un nume format din iniţialele entităţilor asociate.

Descrierea relaţiilor impune şi specificarea *atributelor* asociate, dacă ele există. De exemplu, relaţia *Examen* dintre entităţile Student şi Disciplină are ca atribute proprii *Data_exam* şi *Nota*. Întradevăr, atributul *Nota* reprezintă o proprietate relaţiei dintre un student şi o disciplină la care susţine examen, adică o proprietate a relaţiei stabilite între două instanţe ale celor două entităţi. Dacă *Nota* ar fi atribuită uneia din cele două entităţi, atunci am avea un atribut multivaloare, deoarece un student poate avea mai multe note (pentru fiecare disciplină la care a susţinut examen), iar la o disciplină ar putea exista mai multe note (pentru fiecare student care a susţinut examen la disciplina respectivă). Modul de repezentare grafică a atributelor asociate relaţiilor este prezentat în figura 6.9. Dacă unele notaţii nu utilizează un simbol special pentru relaţii (aşa cum este rombul în figura 6.9) atunci se va introduce în diagramă o entitate asociativă care, pentru moment, nu va avea cheie primară.

Relaţii purtătoare de atribute sunt întâlnite doar în cazul relaţiilor binare de tipul „multe-la-multe" sau a relaţiilor ternare. Relaţiile binare de tipul „unu-la-unu" sau „unu-la-multe" nu pot avea atribute, deoarece cel puţin una dintre entităţile asociate prezintă cardinalitatea maximă „unu", ceea ce înseamnă că eventualele atribute ale relaţiei vor putea fi ataşate unei entităţi fără ambiguitate.

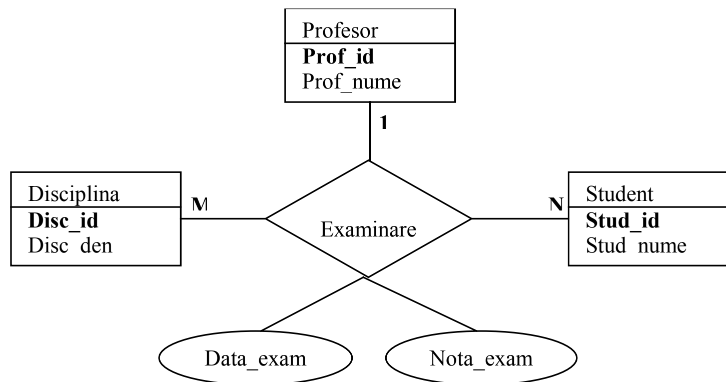


Fig. 6.9 Exemplu de relație purtătoare de atribute

Aplicarea tuturor conceptelor prezentate anterior permit descrierea suficient de clară a tuturor relațiilor între entități identificate în domeniul problemei analizate. O atenție deosebită trebuie acordată **relațiilor redundante**. Două sau mai multe relații sunt considerate redundante atunci când ele sunt utilizate pentru a reprezenta același concept. Un exemplu de relație redundantă este cea dintre *Furnizor* şi *Plata* din figura 6.10. Relația dintre *Furnizor* şi *Plata* este mai curând una indirectă, deoarece se urmăreşte plata facturilor primite. Dacă s-ar ține o evidență globală a datoriilor către furnizori, atunci ar trebui eliminată relația dintre *Factura* şi *Plata*. Indiferent de tipul de evidență, una din cele două relații trebuie eliminată.

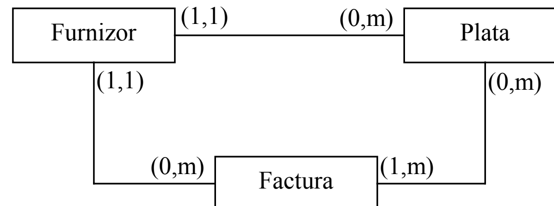


Fig. 6.10 Exemplu de relație redundantă

În schimb, în figura 6.11 nici una dintre cele trei relații nu este redundantă atât timp cât membrii unei asociații pot locui în altă localitate decât cea în care îşi are sediul asociația. Dacă ar exista restricția ca toți membrii unei asociații să locuiască în localitatea de reşedință a asociației, atunci relația dintre *Membru* şi *Localitate* ar fi redundantă.

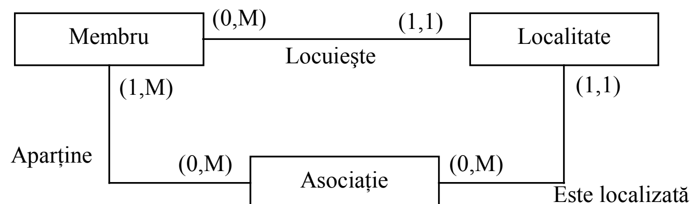


Fig. 6.11 Exemplu de relații neredundante

6.2.4 Construirea diagramelor entitate-relație

Pentru construirea DER se apelează la simbolul dreptunghi, pentru fiecare entitate. Se recomandă ca numele entității să fie redat printr-un substantiv la singular (CLIENT, PRODUS, SALARIAT etc.).

După ce se identifică entitățile, se continuă cu împerecherea lor, fiecare cu fiecare, pentru a descrie ce relații există între ele.

De exemplu, pentru a arăta că o anumită vânzare se referă doar la un client, deşi sunt mult mai multe acte de vânzare, implicit mai mulți clienți, relația poate fi redată schematic astfel:



O vânzare, la rândul ei, constă din unul sau mai multe produse, dar şi produsele pot constitui subiecte ale mai multor vânzări, deci între entitățile VANZARE şi PRODUS se va folosi câte o

săgeată la fiecare capăt al liniei ce leagă cele două entități. Continuăm cu descrierea legăturilor/relațiilor posibile: un PRODUS este în legătură directă doar cu o singură înregistrare STOC, şi invers; ca atare, ele vor fi marcate printr-o linie simplă. Antrenând în sfera discuțiilor şi comanda de aprovizionare, COMANDA, şi furnizorul, FURNIZOR, s-ar putea construi diagrama entitate-relație din fig. 6.12.

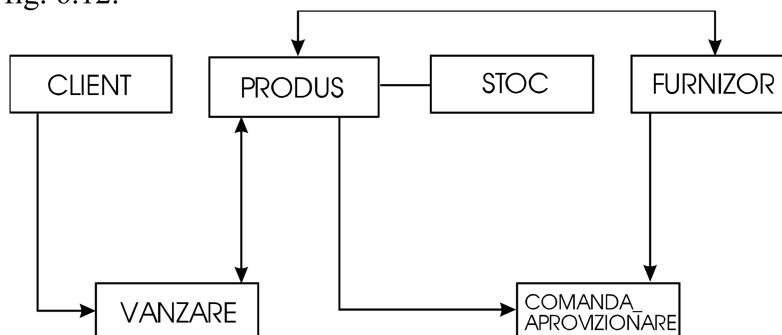
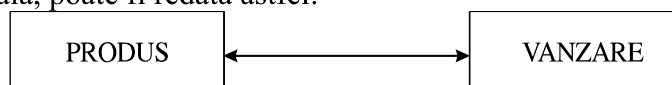
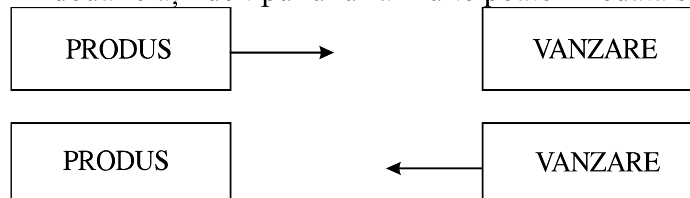


Fig. 6.12 Diagrama entitate-relație pentru operațiunea de vânzare-cumpărare

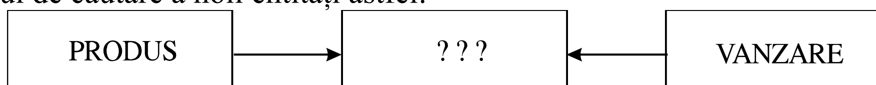
În lucrul cu DER, de o mare importanță, în activitatea de analiză a acestora, este cunoaşterea faptului că întotdeauna *relațiile multe-la-multe pot fi descompuse în două relații de tipul unu-la-multe*, prin aflarea entității-intersecție. Diagrama dintre PRODUS şi VANZARE, de tip multe-la-multe, în forma ei inițială, poate fi redată astfel:



Descompunerea ei în două relații de tipul unu-la-multe poate fi redată sub forma:



Entitatea-intersecție este ceva ce se va impune de la sine între cele două entități, putându-se marca momentul de căutare a noii entități astfel:



Semnele de întrebare înseamnă ce răspuns trebuie să ne găsim la întrebările: *Ce ar putea fi asociat cu entitatea generală simplă PRODUS care să poată da naştere la o legătură de tip* ***multe****?*, precum şi *Ce anume s-ar putea asocia entității simple VANZARE, printr-o legătură de tip* ***multe****, dar care să exprime relația dintre produse şi vânzări?*".

Răspunsul este uşor de dat dacă s-ar folosi un substantiv la plural pentru una din cele două entități cunoscute, deşi normele nu o recomandă. Deci, ce s-ar interpune între o relație de forma PRODUSE-VANZARE? Răspunsul este clar: un articol anume, deoarece vânzarea constă din mai multe articole vândute prin aceeaşi tranzacție externă. Schematic, diagrama va fi:



La fel se poate proceda cu relația multe-la-multe dintre PRODUS şi FURNIZOR, descoperindu-se o nouă entitate, SURSA_PRODUS, care exprimă faptul că un furnizor poate fi sursa de livrare pentru orice produs, la un preț dat şi la un anumit moment contractual.

Relațiile unu-la-unu, de asemenea, sunt supuse analizei pentru a putea constata dacă într-adevăr cele două entități simple sunt total diferite şi nu pot fi combinate în nici un mod. În exemplul anterior, există doar o singură relație unu-la-unu, cea dintre PRODUS şi STOC. Dacă entitatea PRODUS va prelua elementele din STOC, atunci cea de-a doua poate să dispară. Este cazul tipic al nomenclatoarelor, care pot prelua şi elemente ale unor fişiere de stare.

În urma analizelor efectuate asupra diagramei entitate-relație, din fig. 6.12, şi a modificărilor aduse, rezultă o nouă diagramă (fig. 6.13):

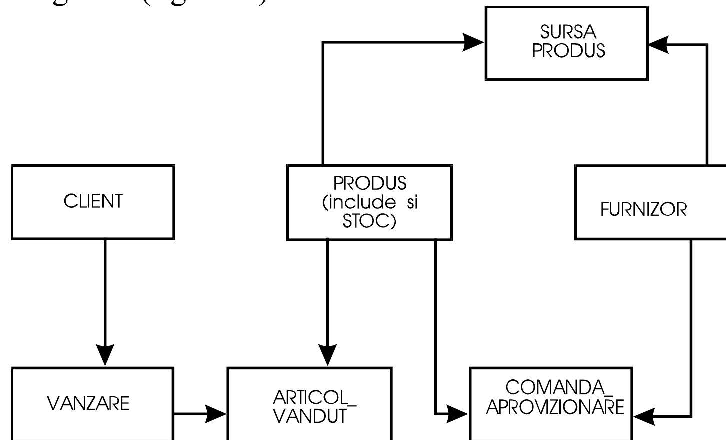
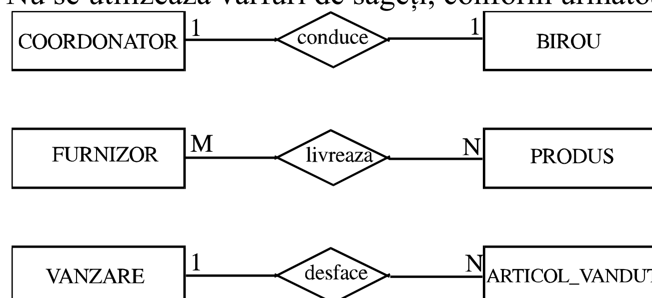


Fig. 6.13 Noua DER pentru operațiunea de vânzare-cumpărare

6.2.5 Notații folosite pentru construirea DER

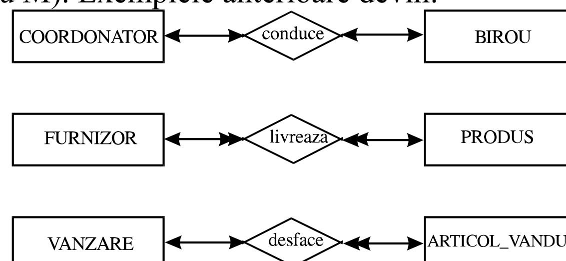
Notația Chen

P.P.Chen este considerat creatorul modelului diagramei entitate-relație (DER), care acordă o importanță deosebită relației ce se poate stabili între două entități, prin folosirea unui simbol special, rombul, pentru redarea acesteia, iar pentru tipul relației (1:1, 1:Multe, Multe:Multe) apelează la caracterele 1, M, N, plasate chiar pe liniile de legătură dintre entitate şi relație sau dintre relație şi entitate. Nu se utilizează vârfuri de săgeți, conform următoarelor exemple:

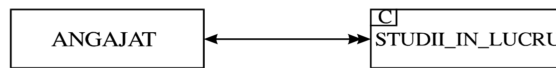


Notația Ross

Ca şi Chen, Ross foloseşte simbolul romb pentru a reprezenta relația sau asocierea, în schimb renunță la caracterele 1, M, N pentru redarea tipului acesteia, apelând la vârful de săgeată unic (pentru 1) sau dublu (pentru M). Exemplele anterioare devin:

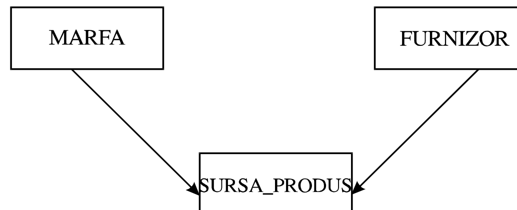


Un aspect nou îl reprezintă şi tratarea diferențiată a tipurilor de entități, Ross recunoscând ca tipuri: **entitățile de fond** (de bază), care sunt de sine stătătoare (CLIENT, ANGAJAT, PRODUS, CLADIRE ş.a.) şi **entitățile-caracteristică**, numite şi **entități dependente**, care nu pot exista fără o entitate de fond (ar fi cazul cont clienți, repartizări-pe-locuri-de-muncă ş.a.). Pentru a le diferenția, Ross foloseşte o notație specială: în colțul din stânga sus al entității-caracteristică delimitează o casetă mică în care plasează litera **C**:



Notația LBMS

Combinația de litere provine de la numele furnizorului produsului CASE AUTOMATE-PLUS, acesta fiind Learmonth and Burchett Management Systems (LBMS). Autorii produsului folosesc doar relații de tipul unul-la-multe (1:M) în construcția DER, pe care ei le numesc Logical Data Structures (LDSs), structuri logice ale datelor. Ca atare, orice relație de tip multe-la-multe (M:N) este descompusă în relații unu-la-multe (1:M), apelând la entitatea intersecție. Edificator este exemplul:



Notațiile folosite sunt:

- linia simplă, fără vârf de săgeată, înseamnă “1”;
- linia finalizată cu un singur vârf de săgeată înseamnă “M” (multe);
- relație opțională este marcată printr-un cerculeț plasat la mijlocul liniei ce uneşte două entități (—○→).